Where to Add Custom Code

Introduction

Welcome to the final lesson in the Divi Theme Workflow module, where I'd like to discuss all the different ways we can add custom code inside our Divi websites.

The first option is Divi itself. It allows us to add custom code in a few different ways, and I will go over each of the possibilities. Next, we can add custom code via a child theme. We can also create a simple functionality plugin, which I'll show you how to do step-by-step. And finally, we can use a plugin like Code Snippets, which can help us organize our work with different bits of custom code.

So these are basically the main options we have. And I'll look into each one in detail, but first, we need to be a bit more specific and narrow down the scope of this lesson. So when I say "custom code", that can mean really anything, but it basically would be some code from one of these three categories: CSS, JavaScript, or PHP.

CSS

Let's start with CSS and consider our options.

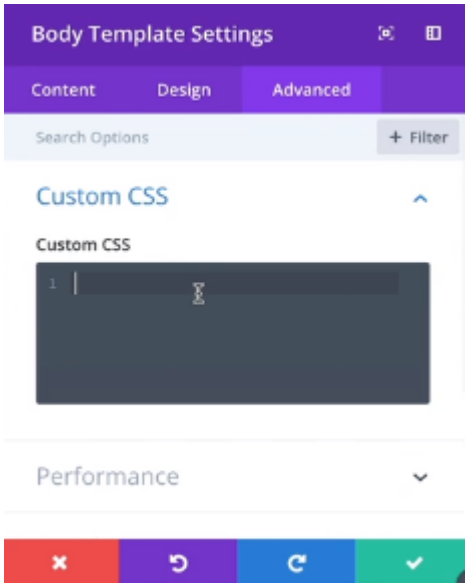
Custom CSS in Divi

Divi has the Custom CSS box, which is accessible both from the Theme Options as well as from the Theme Customizer. And any code we include here will be saved in the database, not in the theme files. So we won't need to worry about Divi updates, and our CSS customizations are safe here. And this code will run on each page of your website. It is a site-wide code.

CSS in Page/Template settings

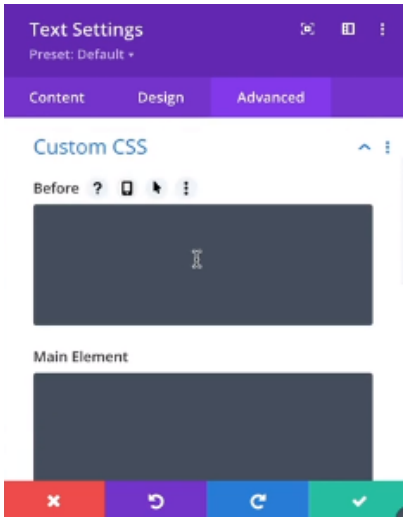
Now, if you have a CSS bit you only need to use in one particular place, you can use the Divi Builder to add it. First, Page Settings (or Template Settings if you are inside the Theme Builder).

In the Advanced tab, we have another Custom CSS box and we can use it to add our CSS. And this code will only be loaded on a page it has been added to, or on a page where the Theme Builder template is active.

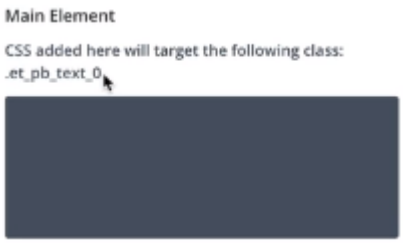


CSS in Divi Builder's element's settings

Next option, Divi Builder element settings. Inside each section, row or module, we can always go to the Advanced tab and use one of these predefined elements, predefined targets. When we get to the lesson about CSS syntax, I will be back here, so that you can see the difference.



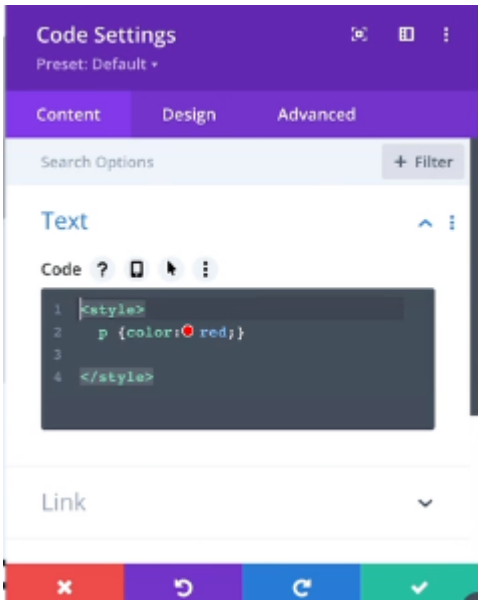
But basically, the CSS code used here only needs to include CSS properties and values. Our targets are defined for us.



So we can write different CSS for different screen sizes or target the hover state, but we cannot use custom media queries to target specific screen width, and we cannot define our own target element, which is kind of limiting. But for a quick, small adjustment, it is sometimes useful.

CSS in a Code module

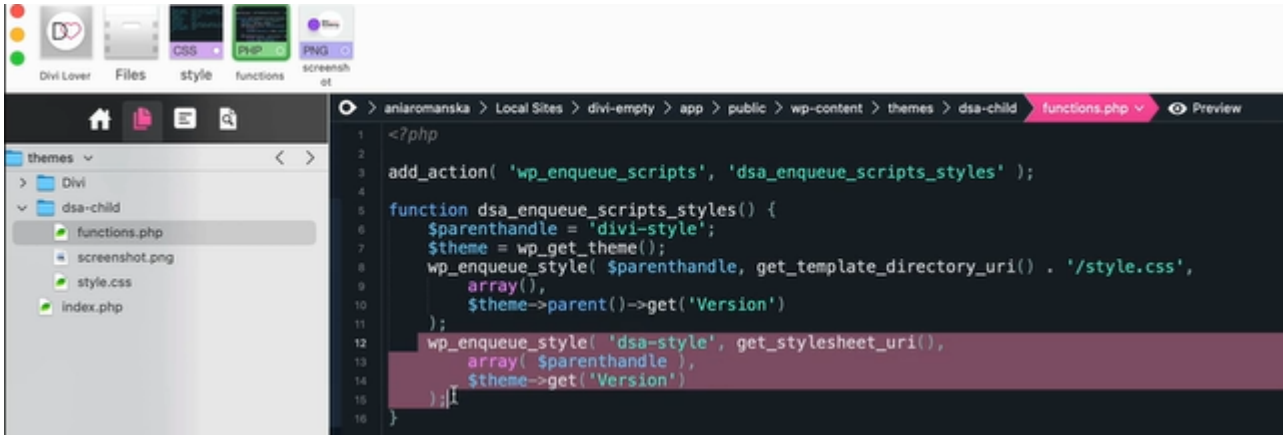
And the final way we can add CSS using the Divi Theme is through the Code module. Inside the Code module, we would only need to specify the <style> tag and we can add our CSS code inside. And similarly to the Page Settings, this bit of code will only work on the page where the Code module is displayed.



CSS in a child theme stylesheet

Next, we have our Child Theme style.css file. I do like to work with a child theme because my CSS files tend to be rather long with hundreds and hundreds of lines of code and managing it through the Theme Options box is not very convenient. I would still use it for a quick edit, but my main customizations are usually inside the child theme stylesheet.

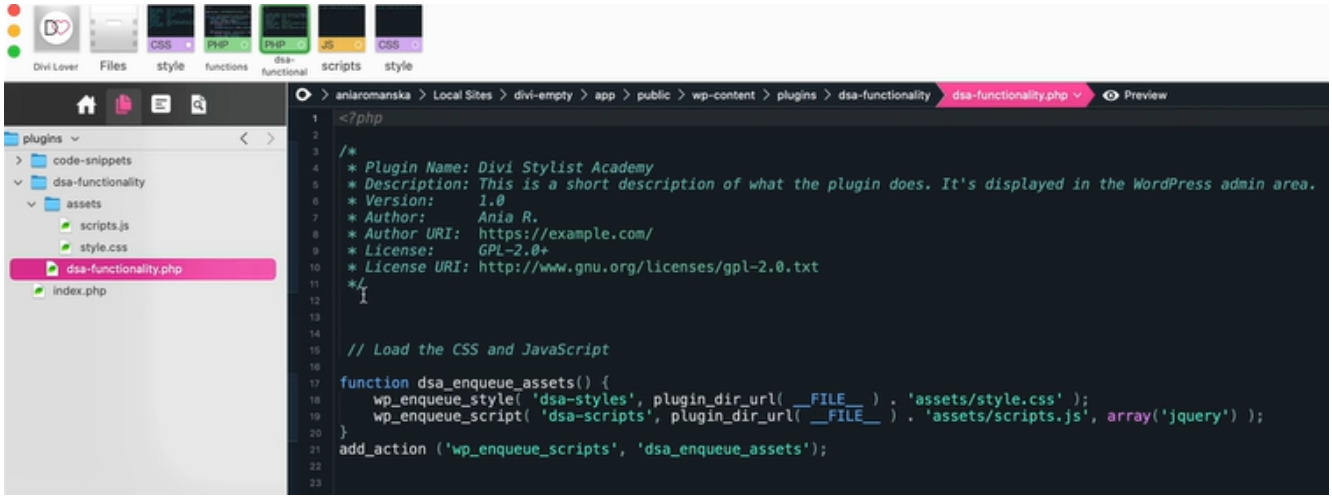
And your child theme can include the enqueue additional CSS files, not only the main style.css.



You can also use **WordPress Conditional Tags** to load it in specific places only. I will refer you to the WordPress Codex for more details on that, but it would be something like: if it's page something, or if any of these conditions - then enqueue "this and this" CSS file, right? So just some basic conditional logic.

Creating a Functionality Plugin

And the same exact approach works within your custom Functionality Plugin, which you can create. And building a simple plugin is very straightforward and similar to creating a child theme. You only really need a PHP file with a proper PHP definition at the beginning, similar to the beginning of the child theme stylesheet. We just need to tell WordPress how to handle that file. And I will include this example in the "Downloads" section but let me explain what is happening here.



First, we are defining the plugin name and description, which will be visible in the Plugins dashboard. Again, a nice way to display your client's name and impress them with their own custom plugin. And further down, we can include any PHP code we want, just like inside the functions.php file in the child theme.

But this simple plugin I have here basically loads a single CSS file and a single custom JavaScript file. It will load it site-wide and you can include your CSS code inside this file. If the plugin is activated in the WordPress dashboard, the CSS customization will apply.

And similarly to the child theme, we can either upload the plugin folder to our website through FTP or zip it and install the ZIP just like any other standard WordPress plugin - you can download the sample ZIP file from the "Downloads" section.

And I think these are basically all the ways to include custom CSS in Divi you might need.

JavaScript

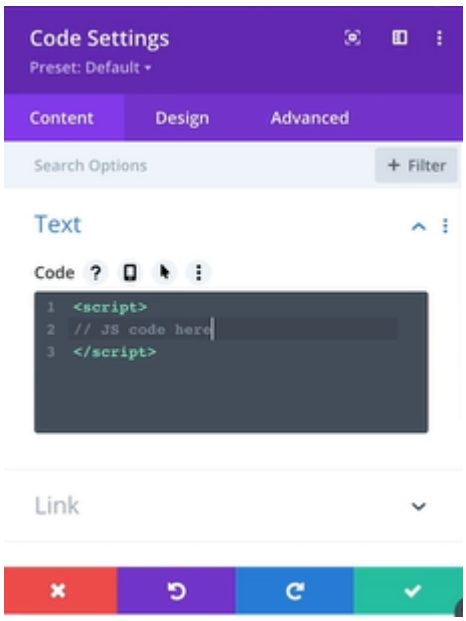
And next off, we have JavaScript.

JavaScript in Divi Theme Options

The first option is Divi Theme Options -> Integration tab. We can include a <script> tag and insert custom code right inside here. Or we can link to an external file. We can decide to include the code inside the head, body, or only on blog posts.

JavaScript in a Code module

We can also use the Code module and insert the same code anywhere using the Divi Builder on the specific page or in one of the Theme Builder templates.



JavaScript in a child theme or functionality plugin

We can also enqueue the JavaScript file inside our child theme or inside the functionality plugin. We only need to call the **wp_enqueue_script** function and define the script name and path.

```
function dsa_enqueue_assets() {
    wp_enqueue_style( 'dsa-styles', plugin_dir_url( __FILE__ ) . 'assets/style.css' );
    wp_enqueue_script( 'dsa-scripts', plugin_dir_url( __FILE__ ) . 'assets/scripts.js', array('jquery') );
}
add_action ('wp_enqueue_scripts', 'dsa_enqueue_assets');
```

And if it needs to be loaded after jQuery, if we are using custom jQuery, the parent jQuery file has to be loaded first, so we can include this dependency as well. And again, WordPress Codex is the best place to learn exactly how to use all these WordPress functions. And you will see the example inside the sample functionality plugin.

PHP

And finally, PHP modifications. We cannot use Divi Theme Options to include custom PHP code, and the Code module cannot work with custom PHP. The only way to add some PHP function to a Divi website is via functions.php file in the child theme, inside our functionality plugin PHP file, or using the Snippets plugin.

But if your PHP modifications include creating custom templates, custom theme templates, or overwriting parent theme PHP files with your custom version - then these types of files should be placed within your child theme.

And I hope this overview gives you a bit more clarity on how you can work with custom code and Divi. Thank you for watching and see you in the next module.

Resources

GET INSPIRED:

Code Snippets plugin

Sample Functionality Plugin - available in this lesson's Downloads section

Action Items

- [ ] Download and edit the sample functionality plugin. Include the PHP code snippet from the **Divi Library** lesson to create a plugin which displays Divi layouts as shortcodes.